Polynomial Regression

Contents

Introduction

Polynomial regression is a tool that allows us to understand and predict the behavior of complex data. Unlike linear regression, which assumes a linear relationship between the independent and dependent variables, polynomial regression can model nonlinear relationships by incorporating polynomial terms.

First, we consider a univariate model:

$$y_i = f(x_i) + \varepsilon_i, \quad i = 1, \ldots, n, \tag{1}$$

In polynomial regression, f is a polynomial function of degree q, expressed as:

$$f(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_q x^q$$

where β_j are the coefficients for $j = 0, \ldots, q$. The model then can be represented as:

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \cdots + \beta_q x_i^q + \varepsilon_i, \quad i = 1, \ldots, n.$$

The coefficients β are estimated using linear regression, considering the model $y = X\beta + \varepsilon$, where $X = [1, x, x^2, \ldots, x^q]$ and $\beta = [\beta_0, \beta_1, \beta_2, \ldots \beta_q]^T$.

Simulating data

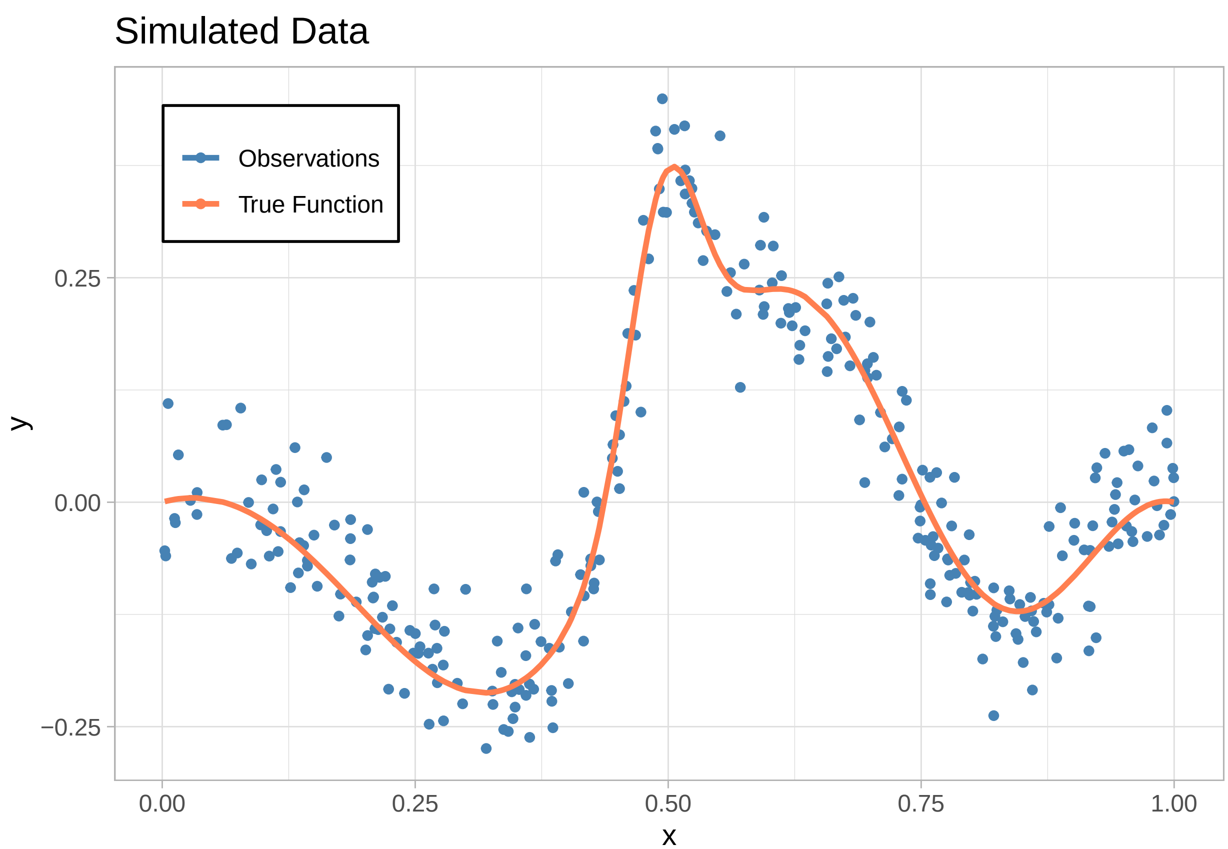
For our examples, we will simulate data to explore how well polynomial regression can fit oscillating values.

```r
# Generate X values
x <- runif(300, min=0, max=1)
x <- x[order(x)]

# Error term
eps <- rnorm(300, mean=0, sd=0.05)

# True and observed Y values
y.true <- -x * cos(5 * (x + 0.5)^2) * (x - 1) +
          1.2 * exp(-(20^2) * (x - 0.5)^2) * (x - 1)^2
y.obs <- y.true + eps
```

To visualize the data, we utilize `ggplot2`:

```r
library(ggplot2)
```

```r
data <- data.frame(x, y.obs, y.true)
ggplot(data = data) +
  geom_point(aes(x = x, y = y.obs, color = "Observations"),
             shape = 20, size = 2) +
  geom_path(aes(x = x, y = y.true, color = "True Function"), linewidth = 1) +
  scale_color_manual(values = c("Observations" = "steelblue",
                                "True Function" = "coral")) +
  labs(title = "Simulated Data",
       x = "x",
       y = "y",
       color = NULL) +
  theme_light() +
  theme(plot.title = element_text(size = 14),
        legend.position = c(0.15, 0.85),
        legend.background = element_rect(fill = "white",colour = "black"))
```




```r
# ggsave("graphs/PolynomialRegression.pdf", width = 10, height = 6, dpi = 300)
```

Fitting the Data

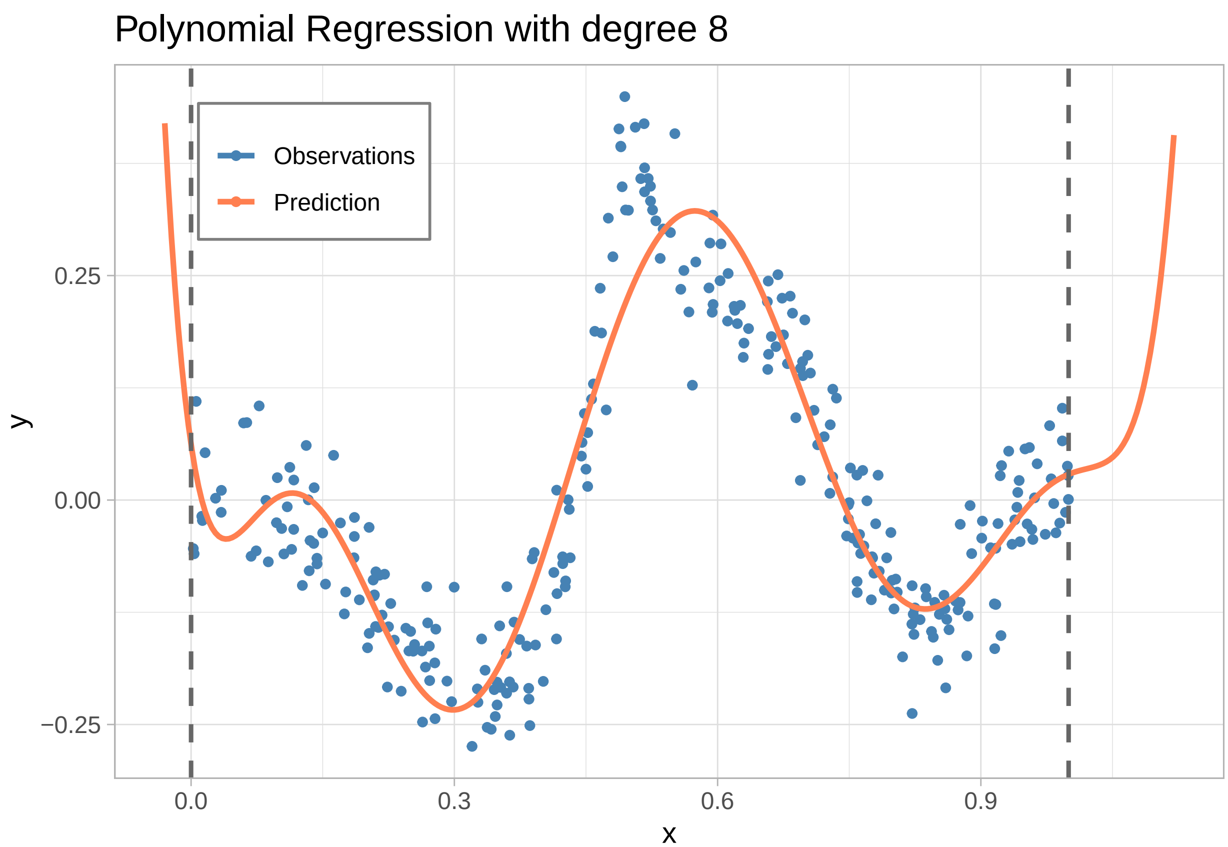
We fit the data using the `lm` function from the `stats` package, starting with a polynomial of degree 8.

```r
polynomial <- lm(y.obs ~ poly(x, degree=8, raw=TRUE))
x.pred <- seq(-0.03, 1.12, length.out = 300)
```

```r
y.pred <- predict(polynomial, newdata = data.frame(x = x.pred))
```

The results are displayed as follows:

```r
data.polynomial <- data.frame(x, y.obs, x.pred, y.pred)
ggplot(data = data.polynomial) +
  geom_point(aes(x = x, y = y.obs, color = "Observations"),
             shape = 20, size = 2) +
  geom_path(aes(x = x.pred, y = y.pred, color = "Prediction"), linewidth = 1) +
  geom_vline(xintercept = 0, linetype = "dashed", color = "gray40",
             linewidth = 0.8) +
  geom_vline(xintercept = 1, linetype = "dashed", color = "gray40",
             linewidth = 0.8) +
  scale_color_manual(values = c("Observations" = "steelblue",
                                "Prediction" = "coral")) +
  labs(title = "Polynomial Regression with degree 8",
       x = "x",
       y = "y",
       color = NULL) +
  theme_light() +
  theme(plot.title = element_text(size = 14),
        legend.position = c(0.18, 0.85),
        legend.background = element_rect(fill = "white", colour = "gray50"))
```




```r
# ggsave("graphs/PolynomialRegression1.pdf", width = 10, height = 6, dpi = 300)
```

In this graph, we observe that the high degree allows the model to accurately fit the data within the variable

X range (in this case, $[0, 1]$). However, outside this range, the model predictions increase rapidly, which illustrates the potential risks of using high-degree polynomials without appropriate constraints. Now, we will explore the effects of using different polynomial degrees:

```
poly.3  <- lm(y.obs ~ poly(x, degree=3, raw=TRUE))
poly.5  <- lm(y.obs ~ poly(x, degree=5, raw=TRUE))
poly.10 <- lm(y.obs ~ poly(x, degree=10, raw=TRUE))
poly.20 <- lm(y.obs ~ poly(x, degree=20, raw=TRUE))
```

We can examine the coefficients and other statistical details by using:

```
summary(poly.3)
```

```
##
## Call:
## lm(formula = y.obs ~ poly(x, degree = 3, raw = TRUE))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.29222 -0.12579 -0.00821  0.10024  0.40436
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      -0.02116    0.03963  -0.534  0.59378
## poly(x, degree = 3, raw = TRUE)1 -0.82190    0.31472  -2.612  0.00947 **
## poly(x, degree = 3, raw = TRUE)2  3.17293    0.69780   4.547 7.94e-06 ***
## poly(x, degree = 3, raw = TRUE)3 -2.50773    0.44678  -5.613 4.58e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1469 on 296 degrees of freedom
## Multiple R-squared:  0.1958, Adjusted R-squared:  0.1877
## F-statistic: 24.03 on 3 and 296 DF,  p-value: 6.059e-14
```

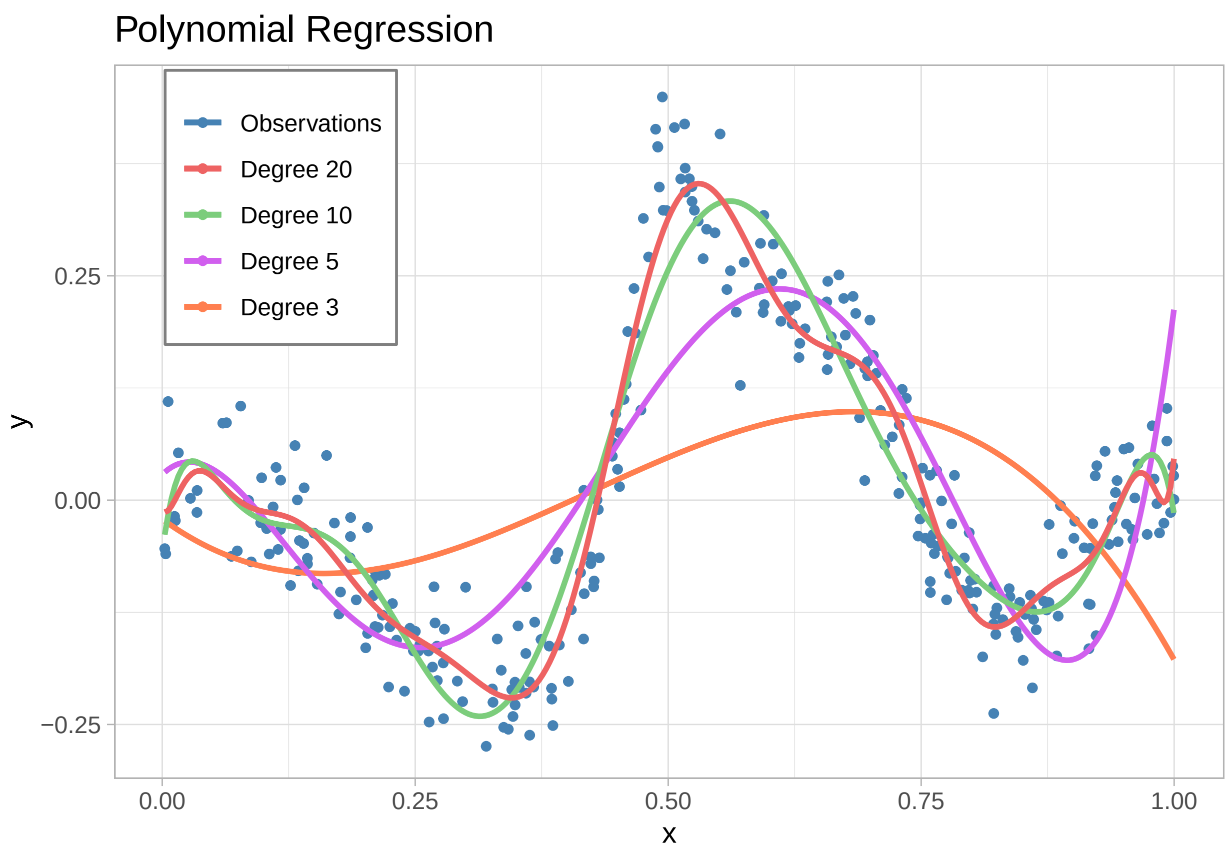
Next, we make predictions using these models:

```
x.pred <- seq(min(x), max(x), length.out = 300)
y.pred3  <- predict(poly.3, newdata = data.frame(x = x.pred))
y.pred5  <- predict(poly.5, newdata = data.frame(x = x.pred))
y.pred10 <- predict(poly.10, newdata = data.frame(x = x.pred))
y.pred20 <- predict(poly.20, newdata = data.frame(x = x.pred))
```

The visualizations of these predictions are as follows:

```
data_pred <- data.frame(x, y.obs, x.pred, y.pred3, y.pred5, y.pred10, y.pred20)

ggplot(data = data_pred) +
  geom_point(aes(x = x, y = y.obs, color = "Observations"),
             shape = 20, size = 2) +
  geom_path(aes(x = x.pred, y=y.pred3, color = "Degree 3"), linewidth = 1) +
  geom_path(aes(x = x.pred, y=y.pred5, color = "Degree 5"), linewidth = 1) +
  geom_path(aes(x = x.pred, y=y.pred10, color = "Degree 10"), linewidth = 1) +
  geom_path(aes(x = x.pred, y=y.pred20, color = "Degree 20"), linewidth = 1) +
  scale_color_manual(values = c("Observations"="steelblue",
                                "Degree 3"="coral",
                                "Degree 5"="mediumorchid2",
                                "Degree 10"="palegreen3",
```

```
                        "Degree 20"="indianred2"),
            limits = c("Observations", "Degree 20", "Degree 10",
                        "Degree 5", "Degree 3")) +
  labs(title = "Polynomial Regression",
       x = "x",
       y = "y",
       color = NULL) +
  theme_light() +
  theme(plot.title = element_text(size = 14),
        legend.position = c(0.15, 0.8),
        legend.background = element_rect(fill = "white", colour = "gray50"))
```




```
# ggsave("graphs/PolynomialRegression2.pdf", width = 10, height = 6, dpi = 300)
```